**INVENTORY MANAGEMENT**

**A MINI-PROJECT BY:**

**SANDHIYA .P 230701282**

**NIRANJANA .K 230701400**

***in partial fulfilment of the award of the degree***

***OF***

***BACHELOR OF ENGINEERING***

**IN**

**COMPUTER SCIENCE AND ENGINEERING **

**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI An Autonomous Institute**

**CHENNAI**

**NOVEMBER 2024**

**BONA FIDE CERTIFICATE**

Certified that this project **“INVENTORY MANAGEMENT”** is the bona fide work of **“SANDHIYA P, NIRANJANA K”** who carried out the project work under my supervision.

Submitted for the practical examination held on

SIGNATURE

Ms. V. JANANEE

Assistant Professor (SG),

Computer Science and Engineering,

Rajalakshmi Engineering College

(Autonomous),

Thandalam, Chennai-602105

SIGNATURE

Mr. G SARAVANA GOKUL

Assistant Professor (SS),

Computer Science and Engineering,

Rajalakshmi Engineering College

(Autonomous),

Thandalam, Chennai-602105

**EXTERNAL EXAMINER**

**INTERNAL EXAMINER**

**ABSTRACT**

**Inventory Management System:**

The Inventory Management System is a Java-based mini-project designed to optimize and automate inventory tracking and management processes. This application simplifies the management of inventory-related data, improving both accuracy and efficiency while eliminating the need for manual record-keeping. It tracks the movement of products, including stock levels, supplier details, sales transactions, and order histories, all stored securely in a centralized database.

The system integrates Java with a database backend to ensure seamless data storage and retrieval. Authorized users can add, view, update, or delete inventory records, with robust security features implemented through user authentication to ensure that only authorized personnel can modify critical inventory data. The application offers advanced search and filtering capabilities, allowing users to quickly retrieve product information based on specific criteria, such as product name, category, or supplier.

In addition, the system includes automated report generation features, which help managers and staff analyze stock levels, sales trends, and order histories. These reports assist in making informed decisions about restocking, sales forecasting, and optimizing inventory turnover.

By employing modern programming techniques and an intuitive user interface, the Inventory Management System showcases how technology can transform inventory operations. This project serves as a scalable prototype for real-world applications, enhancing administrative efficiency, reducing human error, and providing a reliable, real-time inventory tracking solution.

**TABLE OF CONTENTS**

**1. INTRODUCTION**

1.1. INTRODUCTION

1.2. IMPLEMENTATION

1.3. SCOPE OF THE PROJECT

**2. SYSTEM SPECIFICATION**

2.1. HARDWARE SPECIFICATION

2.2. SOFTWARE SPECIFICATION

**3. ENTITY RELATION MODEL**

3.1. ER DIAGRAM

**4. SAMPLE CODE**

4.1. LOGIN PAGE 4.2. DASHBOARD

4.3. FRUITS AND VEGETABLES PAGE

**5. SNAPSHOTS**

5.1. LOGIN PAGE 5.2. DASHBOARD

5.3. FRUITS AND VEGETABLE PAGE

5.4. DAIRY AND POULTRY PAGE

5.5. CEREALS AND GRAINS PAGE

5.6. SPICES PAGE

5.7. MEAT AND POULTRY PAGE

5.8. IMAGES PAGE

**6. CONCLUSION**

**7. REFERENCES**

**INTRODUCTION**

* 1. **INTRODUCTION**

Inventory Management System is a Java-based application designed to efficiently manage and organize inventory-related information. It streamlines the process of recording, storing, and retrieving details about products, stock levels, suppliers, sales transactions, and purchase orders, ensuring smooth and effective inventory control. The system automates critical inventory tasks such as tracking product movement, monitoring stock levels, and generating reports, enabling businesses to maintain optimal inventory and reduce the risk of overstocking or stockouts. With its user-friendly interface and secure data handling, the system enhances overall inventory management efficiency, providing real-time insights for informed decision-making.

**1.2 IMPLEMENTATION**

The **Inventory Management System** project is implemented using Java Swing for the user interface and MySQL for database management.

**1.3 SCOPE OF THE PROJECT**

This project holds immense potential to revolutionize inventory management by offering a seamless, secure, and intuitive platform for organizing inventory-related information. Its core focus is to provide a centralized, efficient solution that empowers businesses, warehouse managers, and supply chain professionals to easily store, access, and update critical details about products, stock levels, suppliers, sales, and purchase orders, all while ensuring a smooth and professional user experience. The system streamlines inventory processes, reduces errors, and enhances decision-making, ultimately improving operational efficiency and optimizing resource management across various industries.

**SYSTEM SPECIFICATIONS**

**2.1 HARDWARE SPECIFICATIONS:**

PROCESSOR: Intel i5

MEMORY SIZE: 16GB

HARD DISK: 500 GB of free space

**2.2 SOFTWARE SPECIFICATIONS:**

PROGRAMMING LANGUAGE: Java, SQL

FRONT-END: Java Swing

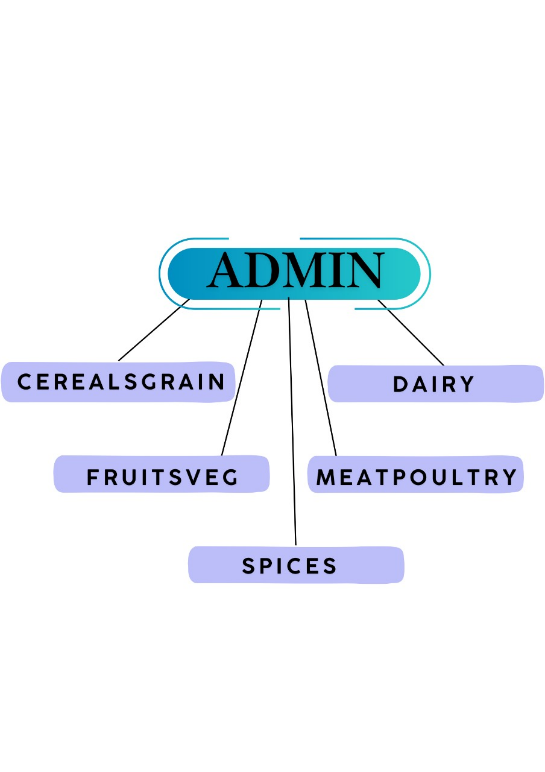
BACK-END: MySQL

OPERATING SYSTEM: Windows 11

**ENTITY RELATION MODEL**

**3.1 ER DIAGRAM**

An Entity Relationship (ER) diagram outlines the structure of the database by representing its entities and their relationships. In this **Inventory Management System**, the Admin can log in to manage key operations, such as adding property details, viewing all property records, and deleting outdated or irrelevant data. In this simplified model, there are no complex relationships between the Property entity and other entities. All essential information, such as property type, location, owner details, and rental status, is directly associated with the property record.

****

**SOURCE CODE**

**4.1 LOGIN PAGE:**

package sem.project2;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class login extends JFrame implements ActionListener {

private JTextField usernameField;

private JPasswordField passwordField;

private JButton loginButton;

private JButton cancelButton;

private Image backgroundImage;

public login() {

setTitle("Login Page");

setSize(700, 400);

setLocationRelativeTo(null);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setResizable(false);

ImageIcon icon = new ImageIcon(getClass().getResource("/images/login.png"));

backgroundImage = icon.getImage().getScaledInstance(getWidth(), getHeight(), Image.SCALE\_SMOOTH);

JPanel backgroundPanel = new JPanel() {

@Override

protected void paintComponent(Graphics g) {

super.paintComponent(g);

g.drawImage(backgroundImage, 0, 0, getWidth(), getHeight(), this);

}

};

backgroundPanel.setLayout(null);

JLabel usernameLabel = new JLabel("Username:");

usernameLabel.setBounds(200, 100, 100, 25);

usernameLabel.setForeground(Color.WHITE);

backgroundPanel.add(usernameLabel);

usernameField = new JTextField();

usernameField.setBounds(300, 100, 200, 25);

backgroundPanel.add(usernameField);

JLabel passwordLabel = new JLabel("Password:");

passwordLabel.setBounds(200, 140, 100, 25);

passwordLabel.setForeground(Color.WHITE);

backgroundPanel.add(passwordLabel);

passwordField = new JPasswordField();

passwordField.setBounds(300, 140, 200, 25);

backgroundPanel.add(passwordField);

loginButton = new JButton("Login");

loginButton.setBounds(250, 200, 90, 25);

loginButton.addActionListener(this);

backgroundPanel.add(loginButton);

cancelButton = new JButton("Cancel");

cancelButton.setBounds(360, 200, 90, 25);

cancelButton.addActionListener(e -> clearFields());

backgroundPanel.add(cancelButton);

add(backgroundPanel);

}

private void clearFields() {

usernameField.setText("");

passwordField.setText("");

}

public void actionPerformed(ActionEvent e) {

String username = usernameField.getText();

String password = String.valueOf(passwordField.getPassword());

if (authenticateUser(username, password)) {

JOptionPane.showMessageDialog(this, "Login successful!");

Dashboard1 dashboard = new Dashboard1();

dashboard.setVisible(true);

dispose();

} else {

JOptionPane.showMessageDialog(this, "Invalid username or password.");

}

}

private boolean authenticateUser(String username, String password) {

String url = "jdbc:mysql://localhost:3306/inventory";

String dbUser = "root";

String dbPassword = "";

boolean isAuthenticated = false;

try (Connection conn = DriverManager.getConnection(url, dbUser, dbPassword)) {

String query = "SELECT \* FROM users WHERE username = ? AND password = ?";

PreparedStatement statement = conn.prepareStatement(query);

statement.setString(1, username);

statement.setString(2, password);

ResultSet resultSet = statement.executeQuery();

if (resultSet.next()) {

isAuthenticated = true;

}

} catch (Exception ex) {

ex.printStackTrace();

}

return isAuthenticated;

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> {

login loginPage = new login();

loginPage.setVisible(true);

});

}

}

**4.2 DASHBOARD DESIGN:**

package sem.project2;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class Dashboard1 extends JFrame implements ActionListener {

private JPanel sidebar;

private JPanel contentPanel;

public Dashboard1() {

setTitle("Dashboard");

setSize(800, 500);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new BorderLayout());

setLocationRelativeTo(null);

sidebar = new JPanel();

sidebar.setPreferredSize(new Dimension(200, 500));

sidebar.setBackground(new Color(1, 54, 94));

sidebar.setLayout(new BoxLayout(sidebar, BoxLayout.Y\_AXIS));

JLabel logoLabel = new JLabel("Inventory", JLabel.CENTER);

logoLabel.setFont(new Font("Arial", Font.BOLD, 18));

logoLabel.setForeground(Color.WHITE);

logoLabel.setAlignmentX(Component.CENTER\_ALIGNMENT);

sidebar.add(Box.createRigidArea(new Dimension(0, 20)));

sidebar.add(logoLabel);

addSidebarButton("Fruits and Vegetables");

addSidebarButton("Dairy Products");

addSidebarButton("Cereals and Grains");

addSidebarButton("Spices");

addSidebarButton("Meat and Poultry");

addSidebarButton("Images");

add(sidebar, BorderLayout.WEST);

contentPanel = new JPanel();

contentPanel.setLayout(new BorderLayout());

add(contentPanel, BorderLayout.CENTER);

addImageToContentPanel();

setVisible(true);

}

private void addSidebarButton(String text) {

JButton button = new JButton(text);

button.setAlignmentX(Component.CENTER\_ALIGNMENT);

button.setMaximumSize(new Dimension(180, 40));

button.setForeground(Color.BLACK);

button.setBackground(new Color(198, 219, 255));

button.setFocusPainted(false);

button.addActionListener(this);

sidebar.add(Box.createRigidArea(new Dimension(0, 20)));

sidebar.add(button);

}

private void addImageToContentPanel() {

ImageIcon imageIcon = new ImageIcon(getClass().getResource("/images/dash.jpg"));

JLabel imageLabel = new JLabel(imageIcon);

Image originalImage = imageIcon.getImage();

Image scaledImage = originalImage.getScaledInstance(500, 280, Image.SCALE\_SMOOTH);

imageLabel.setIcon(new ImageIcon(scaledImage));

imageLabel.setHorizontalAlignment(JLabel.CENTER);

contentPanel.add(imageLabel, BorderLayout.CENTER);

}

@Override

public void actionPerformed(ActionEvent e) {

String action = e.getActionCommand();

if ("Fruits and Vegetables".equals(action)) {

openFVPage();

} else if ("Dairy Products".equals(action)) {

openDPage();

}

else if ("Cereals and Grains".equals(action)) {

openCGPage();

} else if ("Spices".equals(action)) {

openSpicesPage();

}

else if ("Meat and Poultry".equals(action)) {

openMPPage() ;

} else if ("Images".equals(action)) {

openImagesPage();

}

else {

displayContent(action);

}

}

private void openFVPage() {

FruitsVeg Page = new FruitsVeg();

Page.setVisible(true);

}

private void openCGPage() {

CerealsGrains Page = new CerealsGrains();

Page.setVisible(true);

}

private void openMPPage() {

MeatPoultry Page = new MeatPoultry();

Page.setVisible(true);

}

private void openImagesPage() {

JFrame TypeFrame = new JFrame("Images");

TypeFrame.setSize(600, 400);

TypeFrame.add(new images());

TypeFrame.setLocationRelativeTo(this);

TypeFrame.setVisible(true);

}

private void openDPage() {

Dairy Page = new Dairy();

Page.setVisible(true);

}

private void openSpicesPage() {

Spices Page = new Spices();

Page.setVisible(true);

}

private void displayContent(String title) {

contentPanel.removeAll();

JLabel titleLabel = new JLabel(title, JLabel.CENTER);

titleLabel.setFont(new Font("Arial", Font.BOLD, 24));

contentPanel.add(titleLabel, BorderLayout.CENTER);

contentPanel.revalidate();

contentPanel.repaint();

}

}

**4.3 FRUITS &VEGETABLES PAGE:**

package sem.project2;

import javax.swing.\*;

import java.awt.\*;

import java.sql.\*;

import java.util.Vector;

public class FruitsVeg extends JFrame {

private JPanel fruitsvegPanel;

private JTable table;

private JTextField idField, nameField, availabilityField;

private static final String URL = "jdbc:mysql://localhost:3306/inventory";

private static final String USER = "root";

private static final String PASSWORD = "";

public FruitsVeg() {

setTitle("Fruits and Vegetables");

setSize(800, 500);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLocationRelativeTo(null);

fruitsvegPanel = new JPanel(new BorderLayout());

fruitsvegPanel.setBackground(new Color(173, 216, 230));

JPanel formPanel = new JPanel();

formPanel.setLayout(new GridLayout(3, 2, 10, 10));

formPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

formPanel.setBackground(new Color(195, 215, 240));

idField = new JTextField();

nameField = new JTextField();

availabilityField = new JTextField();

Dimension textFieldSize = new Dimension(150, 25);

idField.setPreferredSize(textFieldSize);

nameField.setPreferredSize(textFieldSize);

availabilityField.setPreferredSize(textFieldSize);

formPanel.add(new JLabel("ID:"));

formPanel.add(idField);

formPanel.add(new JLabel("Name:"));

formPanel.add(nameField);

formPanel.add(new JLabel("Availability:"));

formPanel.add(availabilityField);

fruitsvegPanel.add(formPanel, BorderLayout.WEST);

table = new JTable();

loadDataFromDatabase();

JScrollPane tableScrollPane = new JScrollPane(table);

fruitsvegPanel.add(tableScrollPane, BorderLayout.CENTER);

JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 10, 10));

buttonPanel.setBackground(new Color(202, 202, 202));

JButton addButton = new JButton("Add");

JButton editButton = new JButton("Edit");

JButton removeButton = new JButton("Remove");

JButton clearButton = new JButton("Clear");

addButton.addActionListener(e -> addRecord());

editButton.addActionListener(e -> editRecord());

removeButton.addActionListener(e -> removeRecord());

clearButton.addActionListener(e -> clearFormFields());

buttonPanel.add(addButton);

buttonPanel.add(editButton);

buttonPanel.add(removeButton);

buttonPanel.add(clearButton);

fruitsvegPanel.add(buttonPanel, BorderLayout.SOUTH);

table.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

int selectedRow = table.getSelectedRow();

if (selectedRow != -1) {

fillFormFields(selectedRow);

}

}

});

add(fruitsvegPanel);

}

private void loadDataFromDatabase() {

try (Connection connection = DriverManager.getConnection(URL, USER, PASSWORD);

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery("SELECT \* FROM fruitsveg")) {

ResultSetMetaData metaData = resultSet.getMetaData();

int columnCount = metaData.getColumnCount();

Vector<String> columnNames = new Vector<>();

for (int i = 1; i <= columnCount; i++) {

columnNames.add(metaData.getColumnName(i));

}

Vector<Vector<Object>> data = new Vector<>();

while (resultSet.next()) {

Vector<Object> row = new Vector<>();

for (int i = 1; i <= columnCount; i++) {

row.add(resultSet.getObject(i));

}

data.add(row);

}

table.setModel(new javax.swing.table.DefaultTableModel(data, columnNames));

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "Error loading data from database.", "Database Error", JOptionPane.ERROR\_MESSAGE);

}

}

private void fillFormFields(int selectedRow) {

idField.setText(table.getValueAt(selectedRow, 0).toString());

nameField.setText(table.getValueAt(selectedRow, 1).toString());

availabilityField.setText(table.getValueAt(selectedRow, 2).toString());

}

private void addRecord() {

try (Connection connection = DriverManager.getConnection(URL, USER, PASSWORD)) {

String sql = "INSERT INTO fruitsveg (id, name, availability) VALUES (?, ?, ?)";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, idField.getText());

preparedStatement.setString(2, nameField.getText());

preparedStatement.setString(3, availabilityField.getText());

preparedStatement.executeUpdate();

loadDataFromDatabase();

clearFormFields();

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "Error adding record to database.", "Database Error", JOptionPane.ERROR\_MESSAGE);

}

}

private void editRecord() {

try (Connection connection = DriverManager.getConnection(URL, USER, PASSWORD)) {

String sql = "UPDATE fruitsveg SET name = ?, availability = ? WHERE id = ?";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, nameField.getText());

preparedStatement.setString(2, availabilityField.getText());

preparedStatement.setString(3, idField.getText());

preparedStatement.executeUpdate();

loadDataFromDatabase();

clearFormFields();

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "Error editing record in database.", "Database Error", JOptionPane.ERROR\_MESSAGE);

}

}

private void removeRecord() {

try (Connection connection = DriverManager.getConnection(URL, USER, PASSWORD)) {

String sql = "DELETE FROM fruitsveg WHERE id = ?";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, idField.getText());

preparedStatement.executeUpdate();

loadDataFromDatabase();

clearFormFields();

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "Error removing record from database.", "Database Error", JOptionPane.ERROR\_MESSAGE);

}

}

private void clearFormFields() {

idField.setText("");

nameField.setText("");

availabilityField.setText("");

}

public static void main(String[] args) {

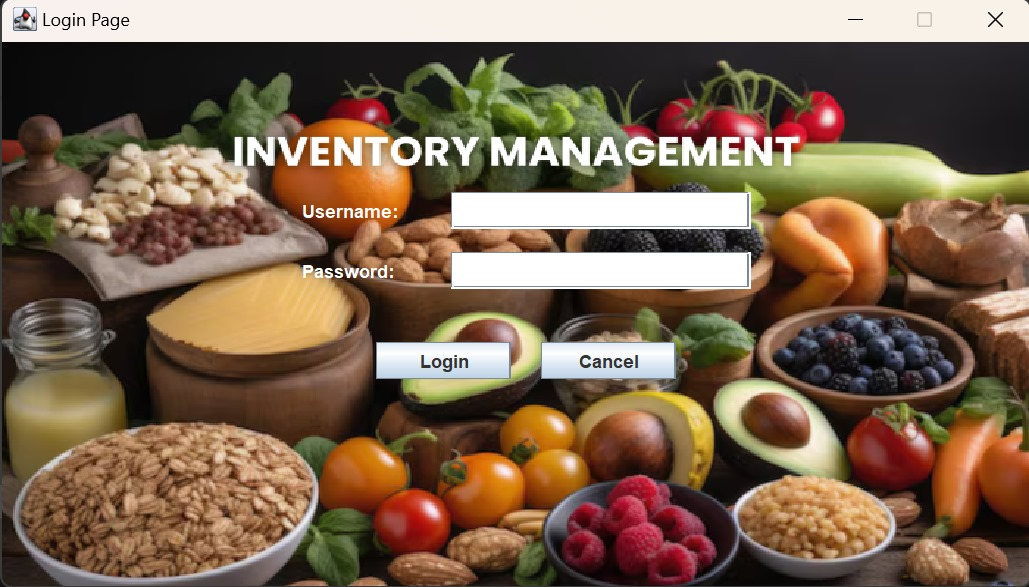
SwingUtilities.invokeLater(() -> new FruitsVeg().setVisible(true));

}

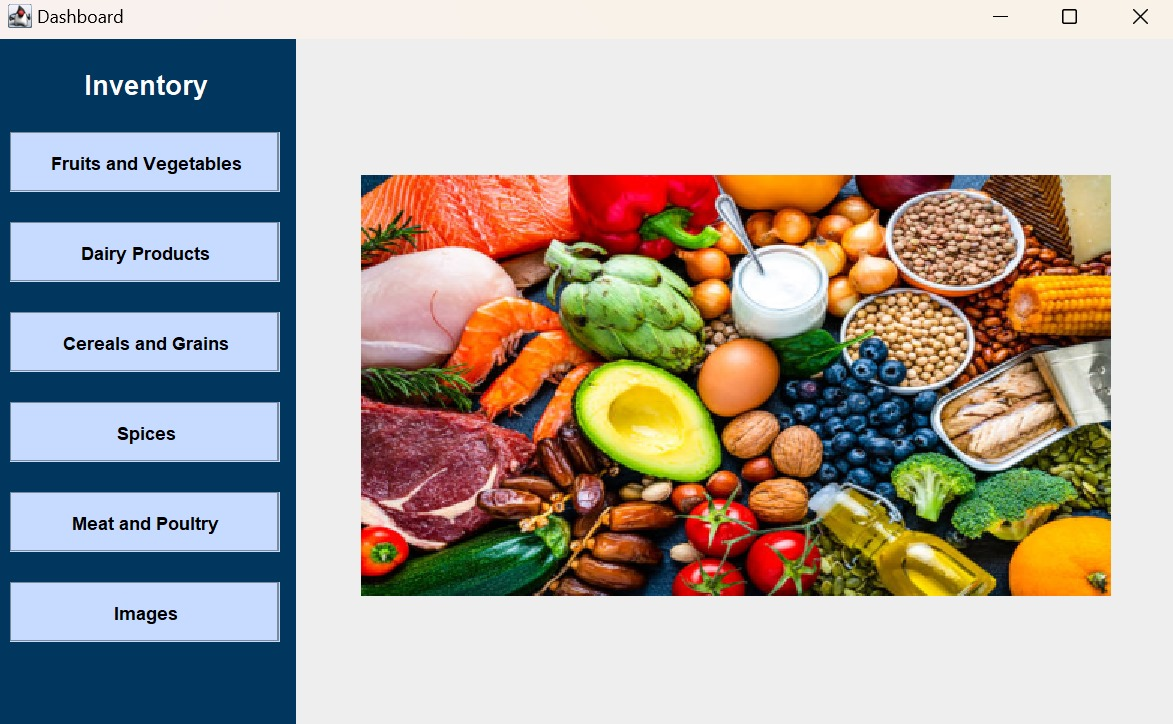
}

**SNAPSHOTS**

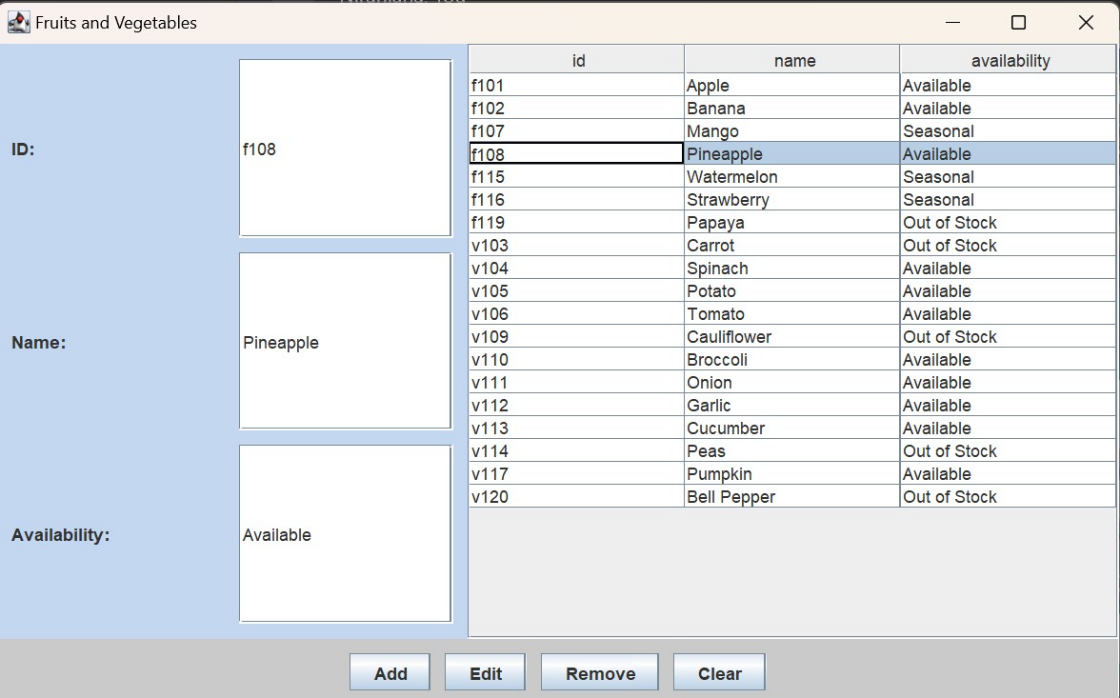
**5.1 LOGIN PAGE:**

****

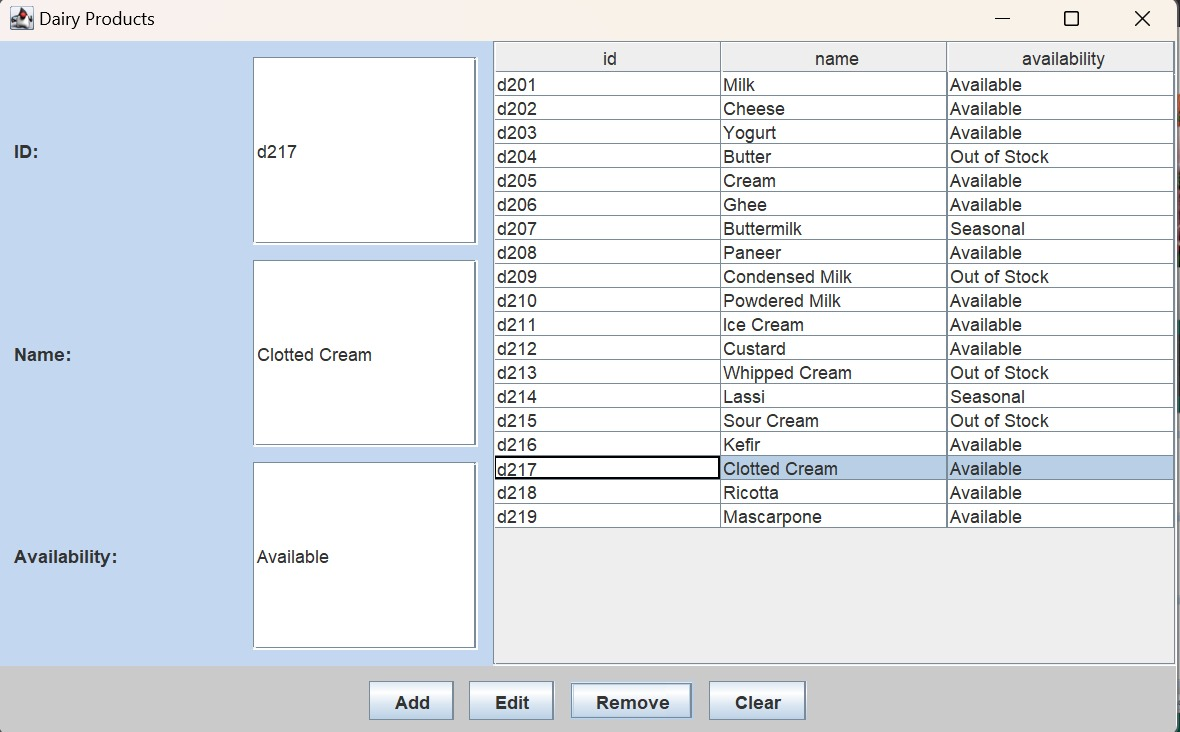
**5.2 DASHBOARD**

****

**5.3 FRUITS AND VEGETABLES PAGE:**

****

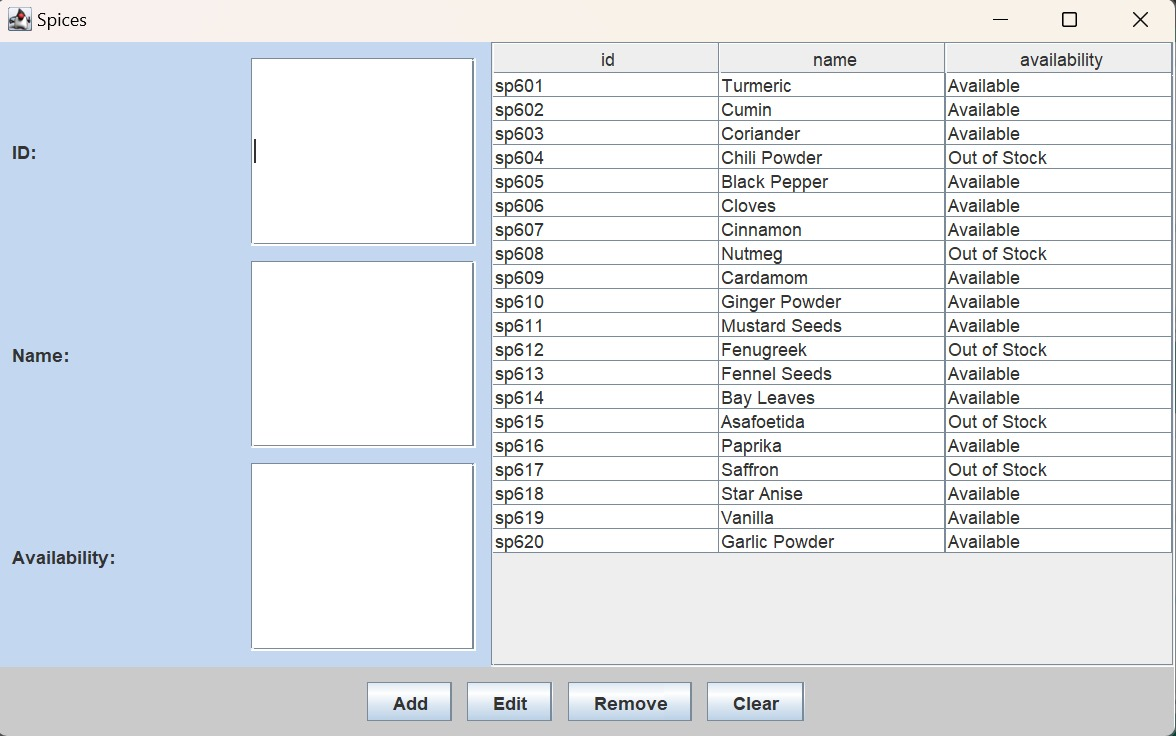
**5.4 DAIRY PAGE**

****

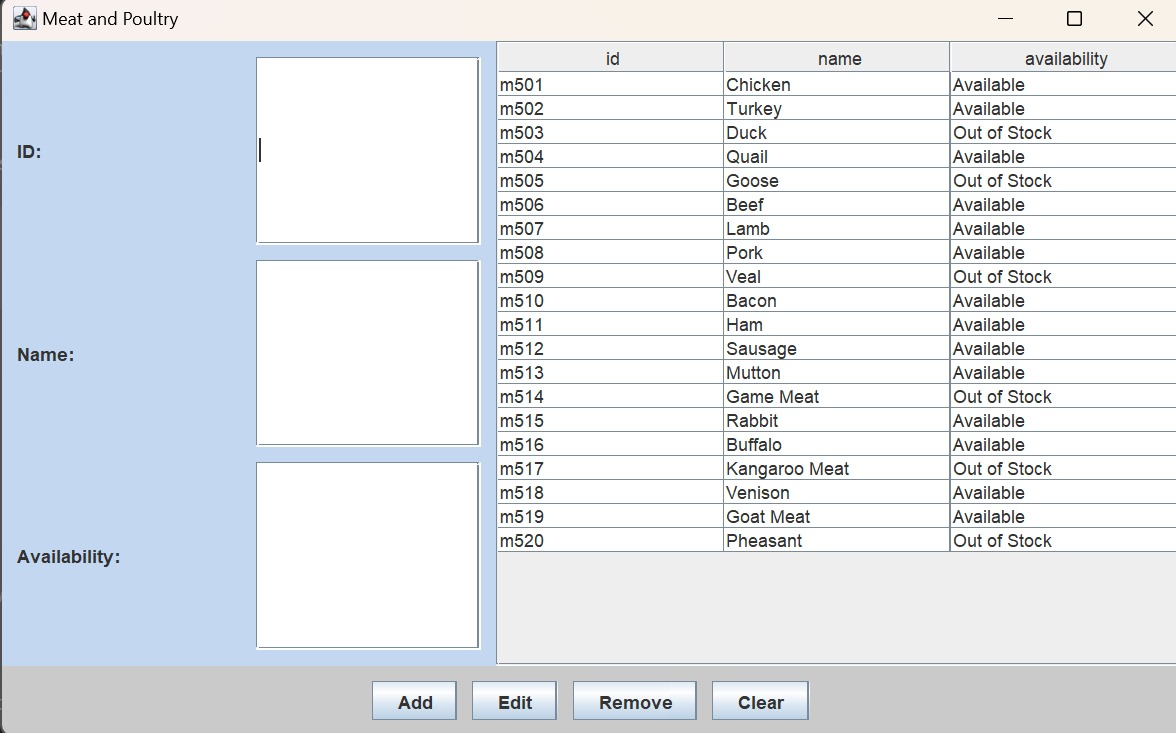
**5.5 CEREALS AND GRAINS PAGE**

****

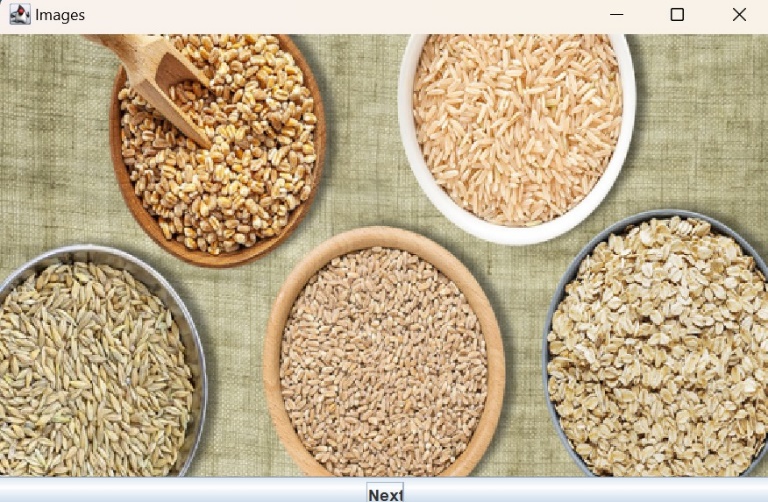
**5.6 SPICES PAGE:**

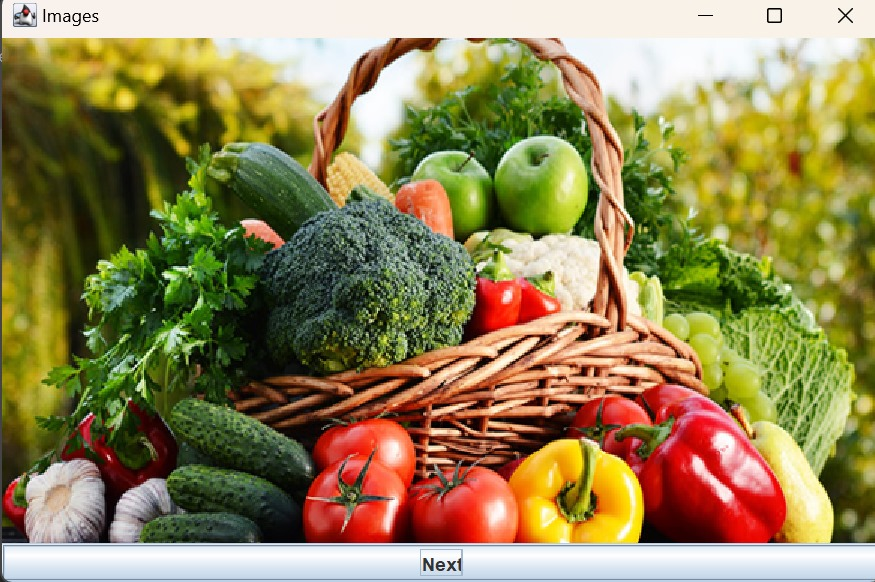
****

**5.7 MEAT AND POULTRY PAGE**

****

**5.8 IMAGES PAGE:**

****

****

****

**CONCLUSION**

The project offers a comprehensive solution for efficient inventory management. By automating various tasks, this system enhances the accuracy and efficiency of inventory operations. It provides a centralized platform for storing, retrieving, and analyzing inventory-related data, enabling informed decision-making and streamlined management. The system features a user-friendly interface for easy data input and retrieval, ensuring a smooth experience for users. With real-time stock tracking and automated report generation, the system helps optimize inventory levels, reduce errors, and improve overall operational efficiency.

**REFERENCES**

**1.** <https://www.javatpoint.com/java-tutorial>  **2.** https://www.wikipedia.org/

**3**. https://www.w3schools.com/sql/